Universität Bielefeld

Faculty of Business Administration and Economics

Working Papers in Economics and Management

No. 06-2014
April 2014

Campus-Management Systeme als Administrative Systeme

Campus Management Systems as Administrative Software Systems

Thorsten Spitta Thomas Grechenig

Henning Brune

Marco Carolla Stefan Strobl

Bielefeld University
P.O. Box 10 01 31
33501 Bielefeld – Germany

www.wiwi.uni-bielefeld.de

Campus-Management Systeme als Administrative Systeme

Thorsten Spitta[1], Thomas Grechenig[2], Henning Brune[3], Marco Carolla[1], Stefan Strobl[2]

Zusammenfassung

Ein von der Politik ausgelöster Bruch in den deutsprachigen Ländern – der sog. Bologna Prozess – löste eine große Nachfrage nach neuen Informationssystemen aus, die die akademischen Prozesse Lehre und Forschung unterstützen konnten. Die Software ist in der notwendigen Qualität am Markt nicht verfügbar. Einige Systeme sind große Pilotprojekte in Universitäten, die die Rolle von Pionieren spielen. Weil die Universitäten – im Gegensatz zu Wirtschaftsunternehmen – wenig Erfahrung mit solchen Projekten und dem Systembetrieb haben, erscheint es hilfreich, die wesentlichen Eigenschaften organisatorischer Informationssysteme grundsätzlich zu betrachten. Nach Lehmanns Definition vor 35 Jahren über Embedded Software haben wir es mit sehr komplexen und großen Systemen zu tun, die in eine Organisation als Umwelt eingebettet sind. Die Komplexität dieser Systeme liegt in der Datenbasis, die von den Benutzern erzeugt und gepflegt wird. Wir diskutieren aus der Sicht dieser originären Daten, welche Funktionen zum Kern eines *Campus-Management Systems* (CaMS) gehören und welche nicht. Z. B. gehören *E-Learning* und *Bibliothek* nicht dazu, benötigen allerdings sichere und effiziente Schnittstellen. Weil CaMS groß und teuer sind, sollten sie evolutionär in die Organisation implementiert werden.

Schlüsselwörter: Campus-Management, Administrative Software, Konzeptuelles Datenmodell

Abstract

Caused by a politically initiated break in German speaking European countries – the so-called Bologna Process – we observe a huge demand for new information systems supporting the academic processes of teaching and research. The software qualitatively demanded is not available on the market. Some systems are large projects of pilot-systems in pioneer universities. Because universities – in contrast to enterprises – have little experience in implementing and operating such systems, it seems to be worth while to examine the essentials of organizational information systems basically. After Lehman's definition of embedded systems 35 years ago, we look at very complex systems, embedded into large organizations. The complexity of such system's software stems from its database, created and maintained by the the organization's users. We argue, from our basic view at original data, which functions are part of the core of a *campus management system* (CaMS) and which are not. E. g. *E-learning* or *library* do not belong to this core, but need secure and efficient interfaces to it. Because CaMS are large and expensive they should be implemented into an organization evolutionary.

Keywords: campus-management, administrative software, conceptual data model

1 Universität Bielefeld, Fakultät für Wirtschaftswissenschaften, Angewandte Informatik. {thSpitta | marco.Carolla@uni-bielefeld.de}
2 Technische Universität Wien, Industrielle Softwaretechnik. {thomas.grechenig | stefan.strobl@inso.tuwien.ac.at}
3 Universität Bielefeld, Dezernat IT/Orga. henning.brune@uni-bielefeld.de

Inhalt

1 Das Problem

Der sog. „Bologna-Prozess“ [27] hat eine große Nachfrage nach zentral betriebener Standardsoftware ausgelöst, die nur auf den *ersten* Blick verfügbar ist [15]. Es werden *Hochschulinformationssysteme* bzw. *Campus-Management Systeme* (CaMS) angeboten und bezüglich der gebotenen Funktionen aufgezählt [2; 55]. Für drängende Fragen zu sehr komplexen und überwiegend neuen Softwaresystemen bieten jedoch Funktionsübersichten keine hinreichende Antwort.

Die zunächst zentralen Fragen dieser Art sind:

- Welche speziellen Anforderungen haben große Hochschulen?
- Welche Typen von Software sind für diese Anforderungen nötig?
- Gibt es bereits Standardsysteme, zwischen denen gewählt werden kann?

Die wichtigsten **Anforderungen** des Bologna-Prozesses waren gestufte Studiengänge (Bachelor und Master), studienbegleitendes Prüfen und eine Modularisierung des Lehrangebots, um mehr individuelle Profile der Abschlüsse zu ermöglichen. Dies erforderte eine sehr starke Erhöhung buchungspflichtiger Vorgänge, die nicht nur transaktionssicher sondern über lange Zeiträume auch *rechtssicher* sein mussten (s. z. B. [32, S. 42]). Die Zweiteilung der alten Abschlüsse brachte einen Zwang zur Prozessbeschleunigung mit sich, da z. B. nur auf der Basis des gerade beendeten Bachelorstudiums über eine Zulassung zum Master entschieden werden konnte.

Diese Anforderungen lassen sich technisch nur mit sog. **Betrieblichen Informationssystemen** [2; 13, S. 10, 15f.; 36; 53] erfüllen, die schon sehr lange nur Datenbank gestützt mit der nötigen Integritätssicherung und Performance betrieben werden können. Grechenig benutzt den „eingebürgerten“ aber irreführenden Begriff *Informationssystem* nicht; er bringt das Thema auf die griffige Formel *Software = Peopleware* [19]. Hierauf gehen wir in Abschnitt 2.4 näher ein. Nur durch eine konsistente Datenbank lässt sich eine Organisation bei vielen Vorgängen pro Zeiteinheit so integrieren, dass schnelle *und* sichere Prozesse möglich sind. Die Integrationswirkung solcher „Informationssysteme“ kommt nur dann zu Stande, wenn die Daten redundanzfrei gehalten werden und alle Organisationseinheiten auf die *selben* Daten schauen und auf Basis der *selben* Daten Berechnungen erstellen oder Entscheidungen fällen.

Nur bei oberflächlicher Betrachtung kann von *existierenden* **Standardsystemen** für Campus-Management gesprochen werden. Bei genauerer Prüfung der angebotenen Produkte lässt sich feststellen, dass es nur *ein* in Deutschland marktbeherrschendes System gibt, das diesen Status beanspruchen kann, allerdings nach Preisgabe wichtiger Ansprüche, die wir oben formuliert hatten. Das den deutschen Markt beherrschende System der HIS GmbH (*Hochschul-Informations-System*) setzt *nicht* auf einer zentralen Datenbasis auf und gilt technologisch als veraltet [8; 45]. Die Programme sind Datei-basierte Bausteine – auch *mit* Datenbanksystem, in ihren Ursprüngen rund 20 Jahre alt und schwer verständlich (s. auch [50, S. 1]). Eine geregelte Release-Pflege findet nicht statt.[4] Für außen Stehende sei ergänzt, dass die Gesellschafter der HIS GmbH die Deutschen Bundesländer und zu ca. 33% der Bund sind.

Die in wenigen großen Universitäten im Echtbetrieb befindlichen Neuentwicklungen [15] lassen hoffen, dass sich konkurrierende Standardsysteme entwickeln (s. auch [3; 8] mit Vergleichen). Sie haben jedoch den Status von Pilot-Anwendungen noch lange nicht verlassen. Die „Pioniere“ dieser Pilot-Anwendungen zahlen einen hohen Preis. Informell bekannt, aber nicht zitierbar, sind mindestens zwei den Autoren bekannte Fehlschläge dieser Projekte mit sehr hohen Kosten für die Auftraggeber bei vergleichbar sehr geringem Nutzen (s. auch [15, S. 32]). Es sollte aber gesehen werden, dass Derartiges auch in der Wirtschaft geschieht, nur seltener öffentlich wird [12].

Der nächste Punkt und auch schon der Begriff *Peopleware* zielen darauf ab, dass man über implementierte Anwendungssoftware nur sinnvoll sprechen kann, wenn die **organisatorische Einbin-**

4 Einer der Autoren hat 15 Jahre dienstliche Erfahrungen mit dem Prüfungsverwaltungs-Modul HIS-POS.

dung jeder einzelnen Softwarekomponente betrachtet wird. Die originären Daten, aus denen jede weitere Information abgeleitet wird, werden von datenverantwortlichen Organisationseinheiten erzeugt, und zwar eineindeutig [48, Kap. 8].

Beim Campus-Management geht dies noch weiter. Einen wesentlichen Teil der *originären* Daten erzeugen die Studierenden, die im Gegensatz zu Systemen der Wirtschaft eben *keine* „Kunden" sind, sondern Systemelemente[5]. Sich überschneidende Änderungsrechte an Daten sind eine Ursache für vielfältige organisatorische Störungen (s. auch [6, S. 14f., 20f.]). Diese elementare Unterscheidung der Datenentstehung ist in wissenschaftlicher Literatur kaum zu erkennen, in der praktischen Datenverarbeitung aber eine Selbstverständlichkeit. Die sog. Praxis spricht von Stamm- und Bewegungsdaten; besser geeignet erscheint für die originären Daten das Begriffspaar *Grund-* und *Vorgangsdaten*. Sie müssen von *abgeleiteten* (= errechnete) Daten unterschieden werden (vgl. hierzu [48, S. 97ff.]). Hier erscheinen die Hochschulen organisatorisch auf dem Stand der Wirtschaft der 80er Jahre. Sie orientieren sich an *Rechenzentren*, selbst wenn diese in „Zentren für Informationsverarbeitung" umgetauft wurden (s. z. B. [7]). Diese leisten wichtige Dienste für die Fakultäten und Einheiten. Sie würden ein zentrales Campus-Management System u. E. auch zuverlässig betreiben. Aber komplexe soziotechnische Projekte durchführen und die Evolution dieser Systeme zu begleiten; damit haben sie weder Erfahrung, noch verfügen sie über das notwendige Personal.

Das Problem, das die sehr vielen Hochschulen als Nachfrager von Standardsoftware also haben, lässt sich folgendermaßen zusammenfassen:

- Die Hochschulen brauchen dringend Campus-Management Systeme,
- die aber in ausreichender Qualität als Standardsoftware (noch) nicht vorhanden sind.

Ökonomisch gibt es eine sehr starke Nachfrage, die aber aus inhaltlichen Gründen nicht schnell befriedigt werden kann, da dem technische und organisatorische Gegebenheiten entgegen stehen.

Wir besprechen die Teilsysteme Lehre und Forschung in den Kapiteln 2 bis 3 und weisen dem "Wissens-Dienst" *Bibliothek* einen zentralen Platz außerhalb des CaMS zu. Danach ziehen wir in Kapitel 4 ein Fazit zu Administrativen Systemen. In Kapitel 5 vertiefen wir den zentralen Kern der Datenbasis eines CaMS, die Studienstruktur. Hierzu wurde die Ankündigung aus [10] umgesetzt, ein Referenz-Datenmodell für Campus-Management Systeme zu entwerfen, das demnächst publiziert wird. Die Validation erfolgte mittels einer Stichprobe von 30 Studiengängen aus 10 der 60 großen deutschsprachigen Hochschulen in Deutschland, Österreich und der Schweiz.

2 Das Teilsystem Lehre

Eine „Big-Bang" Einführung großer soziotechnischer Softwaresysteme ist hoch riskant, insbesondere bei Neuentwicklungen [47]. Daher sollte stufenweise implementiert werden. Unter *Implementierung* verstehen wir den technischen *und* den organisatorischen Vorgang [15; 44, S. 73]. Um die Möglichkeiten und Grenzen einer stufenweisen Einführung zu untersuchen, muss die Abhängigkeit zwischen Funktionen und Datenbasis betrachtet werden. In erster Näherung nehmen wir drei funktional sichtbare Bausteine eines CaMS an – Lehre, Forschung und spezielle Dienste (insbes. Bibliothek) – und beginnen mit dem ersteren. In diesem Teilsystem liegt offenbar seit der Änderung der Hochschulgesetzgebung („BA/MA") auch die höchste Dringlichkeit. Es wird sich zeigen, dass dieses Teilsystem u. a. wegen der Heterogenität seiner Benutzer und hohen Spitzenlasten bei Buchungen das komplexeste ist.

2.1 Datenbasis und Funktionen

Die Integration über eine einheitliche Datenbasis läuft einer evolutionären, modularisierten Einführung zuwider, weil die Komponenten wegen gemeinsamer Daten schnell voneinander abhängig wer-

5 Die Kunden eines Unternehmens sind Elemente der *Umwelt* und nicht des Systems.

den. Um trotzdem eine stufenweise Einführung zu ermöglichen, müssen diejenigen Komponenten gefunden werden, die zwingend zusammen gehören.

Die Antwort lässt sich nur über die Datenbasis geben, bei der die Unterscheidung zwischen Grund- und Vorgangsdaten als originäre und den daraus abgeleiteten sonstigen Daten eine wesentliche Rolle spielt. Tabelle 1 betrachtet sie mit den Funktionen eines CaMS. Die in Spalte 1 gezeigte Funktionsübersicht impliziert eine erste Aussage darüber, was den Kern eines CaMS für die Lehre ausmacht. Zuvor muss aber über die Rolle der Daten gesprochen werden. Die Grunddaten sind starke Entitätstypen, die von anderen nicht abhängig sein dürfen. Ein Grunddatentyp sollte möglichst nur in genau *einem* Teilsystem verwaltet werden. Die Mengengerüste von Grunddaten sind relativ gering, im Gegensatz zu Vorgangsdaten, denn die Masse der Daten entsteht in den „betrieblichen" Vorgängen. Sie sind als schwache Entitätstypen von mindestens einem Grunddatentyp abhängig.

Funktion	***--- originäre Daten ---***		***abgeleitete Daten***
Verwaltung von ...	***Grunddaten***	***Vorgangsdaten***	*Beispiele*
.. Studierenden	Student	Rück-/ Anmeldung	Studienfortschritt
.. Lehrveranstaltungen	Lehrangebot	Teilnahme	Teilnehmer-Statistik
.. Räumen	Raum	Belegung	Belegung
.. Studiengängen	Studiengang	Vorlesungsverzeichnis	Studierenden-Statistik
.. Prüfungen	Leistungs-Regelungen	Leistungs-Exemplar	Zeugnis
.. individuellen Stundenplänen	Stud+La+Stg+Raum	Belegung	Überschneidungsquote

Tab. 1: Funktionen, originäre und einige abgeleitete Daten eines CaMS

Die abgeleiteten Daten wiederum sind überwiegend auf Vorgangsdaten aufgebaut, häufig in verdichteter Form [48, S. 97ff.]. So baut etwa ein Zeugnis beim BA/MA-Lehrkonzept auf einer Vielzahl studienbegleitender Leistungen auf. Auf die sich bereits hier stellende Frage nach einer Abgrenzung zwischen CaMS und **E-Learning** gehen wir in Abschnitt 2.5 ein.

Die letzte Zeile von Tabelle 1 zeigt ein wichtiges Beispiel, das eher *nicht* für zentrale Planer, Dozenten oder Fakultäten relevant ist, sondern für die Stakeholder, für die das Ganze stattfindet, die Studierenden. Bei hoher Wahlfreiheit ist es sehr schwierig und in jedem Einzelfall sogar kombinatorisch unmöglich, die Überschneidungsfreiheit jedes individuellen Stundenplans zu gewährleisten (s. hierzu [9, S. 486ff.]). Bereits in einer mittelgroßen Universität wie Bielefeld werden seit der Einführung von BA/MA über 500 Studiengänge[6] studiert. Das Beispiel zeigt aber auch die Abhängigkeiten der Funktionen von Grunddaten. Wenn Teilsysteme für die ersten vier Grunddaten-Typen nicht existieren, kann die Funktion nicht angeboten werden, weil die originären Daten fehlen.

2.2 Datenbank-Transaktionen

Zum Semesterbeginn und -abschluss treten Transaktionsspitzen auf, die technisch beherrscht werden müssen, insbesondere zum Beginn. Dann belegen die wichtigsten Stakeholder eines CaMS ihre Veranstaltungen, bzw. bilden ihre Stundenpläne – die Studierenden. Dabei werden originäre Vorgangsdaten gebucht. Wenn in solchen Zeitabschnitten technische Probleme auftauchen, weil z. B. Server zu schwach dimensioniert sind, wird die gesamte Organisation in zeitkritischen Arbeitsprozessen behindert.

Auch die Bildung von Teilsystemen bzw. Komponenten muss unter dem Aspekt von Datenbanktransaktionen betrachtet werden. Sie beschränken sich bekanntlich auf *schreibende* Zugriffe, also die Veränderung originärer Daten. Warum dies so ist, wird im nächsten Abschnitt erläutert. Schreibende Operationen bzw. Methoden sollten nicht über die Grenzen von Bausteinen hinaus rei-

6 20 Kern- * 25 Nebenfächer = 500 Studiengänge. Diese haben noch je 3 Profile, was sich bei 4 Pflicht-SWS auf 6.000 nötige Zeitfenster (ZF) ohne Überschneidungen aggregiert. Möglich sind aber pro Student nur 60 ZF/Woche bei 12 Stunden pro Tag (8:00 – 20:00) (Beispiel von Maik Jablonski aus dem Vortrag zu [9]).

chen, auch wenn das technisch mit einem 2-Phasen-Commit möglich wäre. Beim Datenexport gibt es nur *lesende* Zugriffe, die die Integrität einer Datenbasis nicht gefährden können.

2.3 Abgeleitete Daten

Die Unterscheidung in originäre und abgeleitete Daten ist nicht neu [44], aber im Hochschulbereich wenig bekannt. Originäre Daten entstehen in der Umwelt eines Softwaresystems durch menschliche oder maschinelle Akteure. Maschinelle Akteure sind Maschinen, an denen Daten über Sensoren abgegriffen werden, etwa das Lesegerät für eine Mensa-Chipkarte. Die Software ist in die Realität (= Organisation) eingebettet und wirkt auf die Realität zurück, die sie modelliert. Wir werden diesen wichtigen Aspekt in Abschnitt 2.4 vertiefen.

Die originären Daten bilden den Kern einer administrativen Datenbasis und sind auch Gegenstand eines entsprechenden Datenmodells, dessen Kern wir in Kapitel 5 zeigen. Für die Korrektheit sind die Daten erzeugenden Akteure verantwortlich (einzelne oder Organisationseinheiten), die selbstverständlich so gut wie möglich durch Integritätsbedingungen vor Irrtümern geschützt und durch eine lückenlose Abbildung von Prozessen unterstützt werden müssen. Die Akteure der Organisation sind für die Korrektheit der von ihnen erzeugten Daten verantwortlich.

Dies ist bei *abgeleiteten* Daten anders, die aus originären Daten durch *Programme* erzeugt werden. Die Korrektheit des Programms verantwortet ein Programmierer, auch wenn gelegentlich der Benutzer Fachwissen zum Algorithmus oder zu Integritätsbedingungen beisteuern muss. Werden abgeleitete Daten ohne Rückgriff auf gespeicherte Daten erzeugt, verhalten sie sich wie eine mathematische Funktion y = f(x), bei der das Ergebnis beliebig reproduzierbar ist. Setzt eine Funktion auf Zuständen auf (Gedächtnis) – was heute in der Regel der Fall ist – so muss sicher gestellt sein, dass sich die zu Grunde gelegten originären Daten nicht ändern.[7] Dies ist bei Grunddaten selten der Fall, bei Vorgangsdaten jedoch häufig, insbesondere das ständige Entstehen *neuer* Daten in Prozessen mit hoher Frequenz. Dies bedeutet, dass abgeleitete Daten nicht gespeichert werden sollten und heute in vielen Fällen dank der Hardwaretechnik auch nicht gespeichert werden müssen. Die simple betriebswirtschaftliche Formel

$$\mathtt{Bestand}_t = \mathtt{Bestand}_{t-1} + \mathtt{Bewegung}_t$$

führt nur dann zu einem korrekten Bestand, wenn die oben skizzierte Operation in der selben Transaktion ausgeführt wird, wie die Buchung des (originären) Vorgangs *Bewegung*.

Was bedeutet dies für ein CaMS?

(1) Wenn wir Software evolutionär einführen oder entwickeln, muss strikt darauf geachtet werden, dass korrekte und gut archivierte originäre Daten entstehen.

(2) Voneinander existenzabhängige (originäre) Daten können nicht isoliert behandelt werden. Da fast alle relevanten Prozesse Vorgangsdaten erzeugen, können nur diejenigen unterstützt werden, deren Grunddaten mit geklärter Datenverantwortung existieren.

(3) In einer Übergangsphase kann es sinnvoll sein, die dezentrale Erzeugung von Grundaten zuzulassen, aber nur dann, wenn dies Einzelfälle sind und die Daten bei einer Weiterentwicklung kontrolliert integriert werden.

(4) Die Ermittlung manchmal auch komplexer abgeleiteter Daten kann nachgeholt werden, wenn Zeit oder Budget knapp sind.

(5) Gespeicherte abgeleitete Daten sind im Grunde redundant und damit eine mögliche Quelle von Inkonsistenzen.

Ein zeitliches Verschieben „informativer“ Funktionen bei einer evolutionären Einführung kann die Integrität der zentralen Datenbasis nicht negativ beeinflussen. Allerdings sollten sich Entscheider die spätere Einführung eines Data Warehouse Systems nicht einfach und nicht "billig" vorstellen

7 Selbstverständlich benötigt man dann Archive, die die zugehörigen originären Daten für lange Fristen periodenbezogen festhalten, wie dies in Deutschland gem. HGB §§ 238 und 239 (4) für Handelsgeschäfte vorgeschrieben ist.

[41]. Solche Systeme bieten nur einen Nutzen, wenn sie aus einheitlichen Datenquellen gespeist werden [54, S. 485f.] und wenn die laufende Aktualisierung des Datenlagers aus originären Daten sorgfältig spezifiziert und sicher automatisiert wurde.

2.4 *Peopleware* als Eingebettete System

Nach Lehmann [34] entwickeln sich embedded programs mit ihrer Umwelt. Dies zeigt Abbildung 1 mit „R" als Umwelt. Es bedarf für jede Version der Rückkopplung ($t_1..t_n$) einer entsprechenden Spezifikation. Der Begriff *Embedded Systems* wird in der Informatik nur für Software benutzt, die in technische Systeme eingebettet ist und sie überwacht und/oder steuert. Doch auch der Softwaretyp, den wir hier betrachten, kann als „eingebettet" in eine Organisation gesehen werden [44].

Die Akteure der Umwelt des Softwaresystems sind die Benutzer, die originäre Daten erzeugen. Der Begriff *Peopleware* drückt diesen Sachverhalt plastisch aus. Da die Organisation sich im Zeitablauf ändert, wird auch ein „fertiges" Softwaresystem immerzu weiter gepflegt werden müssen. Wir nennen den hier betrachteten Softwaretyp *Administrative Systeme* [44], da die überkommenen Begriffe entweder zu eng oder semantisch falsch und eher von Marketingüberlegungen geprägt sind (*Betriebliches* Informationssystem, Enterprise Resource *Planning* System ERP).[8] Die Systeme sind in Wirklichkeit Buchungssysteme für organisatorische Vorgänge, *keine* Planungssysteme. Aus ihnen werden Informationen für Entscheidungen gewonnen. Selbstverständlich enthalten sie auch Planungsdaten, aber die Vorgangsdaten dominieren die operative Basis.

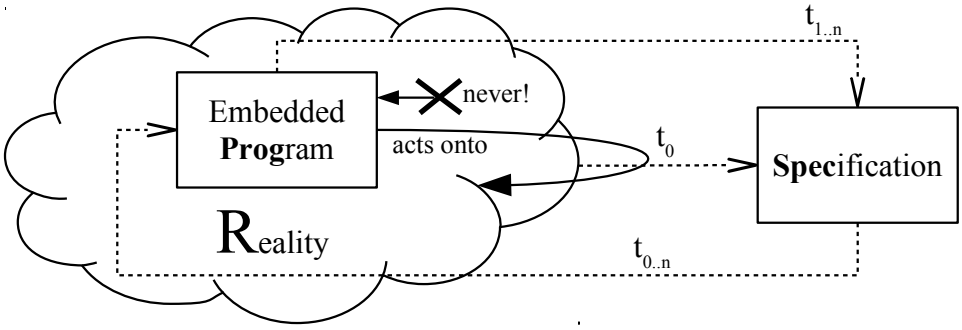

Abb. 1: Administrative Software als *embedded system* (in Anlehnung an [44, S. 31])

Warum haben wir so weit ausgeholt?

- Campus-Management Systeme sind schwergewichtige Administrative Systeme.
- Die wesentlichen Eigenschaften dieser Systeme müssen beachtet werden, damit eine evolutionäre Einführung erfolgreich sein kann, ganz besonders die Abhängigkeiten von Daten.
- Die Systeme müssen in der ersten Stufe möglichst schlank sein. „Schlank" kann bei verschiedenen Komponenten erhebliche Abweichungen haben.

Projektkomplexität aus User- und Interessens-Dynamik						
Nr		Promotors	Core-User	User	Visitors	User-Komplexity
1	Lehrbeispiel	1	2	10	200	0,01
2	Kleines Projekt	1	10	30	3.000	0,25
3	Mittleres Projekt	4	100	2.000	20.000	1
4	Große Hochschule	100	4.000	40.000	1 Mio	100
5	Tickets staatl. Bahn	15	8.000	4 Mio	20 Mio	200
6	ID-Paraguay	50	15.000	8 Mio	> 8 Mio	400

Tab. 2: Komplexität von Softwareprojekten, bewertet mit der Zahl Daten erzeugender User (Grechenig et al. [19, S. 91])

8 Mertens [36] benutzt den Begriff *Betriebswirtschaftlich-administrative Informationssysteme*.

Das zu den Transaktionen zusätzlich „Schwergewichtige“ wird in Tabelle 2 verdeutlicht, die die Positionierung eines CaMS an Hand der Zahl der Promotoren und Benutzer darstellt. Zeile 4 darin zeigt die hohe User-Komplexität eines CaMS, hervorgerufen insbesondere durch die große Anzahl von Promotoren. Die beiden nachfolgenden Projekte (Nr 5 und 6) sind im Vergleich noch komplexere Administrative Systeme als das CaMS. Das in [19] als letztes genannte Beispiel (die hier fehlende Nr 7) mit 1.000 Promotoren und einer User-Komplexität von 4.000 ist u. E. *kein* Administratives System mehr. Es heißt *„E-Health Germany“*.

Ein CaMS ist also technisch (Transaktionen) und organisatorisch *schwergewichtig*. Die Komplexitätsreduktion der Einführung eines solchen Systems ist wünschenswert.

2.5 Abgrenzung CaMS – E-Learning

Allein die eben diskutierte Komplexität des Teilsystems Lehre macht es ratsam, eine Grenze zwischen den administrativen, standardisierbaren Vorgängen der Planung und Abwicklung von Lehre bis zur Prüfung zu ziehen und deren weiter gehender Unterstützung. Erstere gehören in das Teilsystem *Lehre* eines CaMS, letztere können von ergänzenden E-Learning-Systemen oder -Komponenten flexibler geleistet werden. Die Grenze sollte so gezogen werden, dass E-Learning in autonomer Verantwortung des Dozenten optional die *interne* Gestaltung einer Lehrveranstaltung bis zur Abschlussbewertung unterstützt, nicht aber den Gesamtprozess der Lehre von der Anmeldung bis zur abschließenden Bewertung.

Die Grenzen zwischen zentralem CaMS und eher dezentral genutzten E-Learning Systemen sind z. T. fließend, dürfen aber niemals zu einem Nebeneinander redundanter Grunddaten führen. Ein Beispiel ist die Einteilung großer Vorlesungen in Übungsgruppen mit der Abgabe von bewerteten Aufgaben. Dies kann in einer naturwissenschaftlich und technisch geprägten Universität zum zentralen System gehören, in einer eher geistestwissenschaftlich geprägten aber in einem "angedockten" E-Learning-System abgewickelt werden.

Nach der Einführung eines zentralen CaMS können auch generell benötigte Unterstützungsfunktionen mit entsprechenden Daten ergänzt werden, wenn sie Fächer übergreifend benötigt werden. Das auch im Gebiet E-Learning diskutierte Thema Kommunikation wird im Abschnitt 2.7 behandelt. Zuvor muss an zwei wichtigen Beispielen das Thema Abhängigkeit zwischen Anwendungssystemen besprochen werden.

2.6 Abhängigkeiten zwischen Komponenten

Der Datentyp *Dozent* ist ein gutes Beispiel für die Abgrenzung eines CaMS von anderen Anwendungssystemen. Die Grunddaten derjenigen Personen, die ein Entgelt beziehen, müssen in dem System verwaltet werden, das den Datentyp „beschäftigte Person“ mit allen Funktionen verwaltet. Da Lohn- und Gehaltsabrechnung eine typische Standardfunktion ist, wird man hierfür eine Standardsoftware einsetzen und die im CaMS notwendigen Attribute von dort importieren. Die Datenverantwortung für Personaldaten muss bei der zuständigen Organisationseinheit liegen. Denkbar ist wohl eine behelfsmäßige Bezahlung etwa von Tutoren aus dem CaMS, wenn diese Funktion in einem Standardsystem (noch) nicht unterstützt wird. Dies würden wir aber als sog. *Workaround* bezeichnen, den es in vielen Systemen gibt. Es darf nicht vergessen werden, ihn wieder zu beseitigen, denn Workarounds erhöhen Abhängigkeiten zwischen Komponenten.

Eng mit einem Personalsystem verbunden ist eigentlich ein Abbild der Aufbauorganisation. Sie ist jedoch in gängigen Standardsystemen wie etwa SAP ERP so stark mit dem Gesamtsystem verbunden, insbes. der Kostenverantwortung, dass es nicht geraten erscheint, diese Strukturen für ein CaMS zu nutzen, wenn aus dem ERP-System nur Finanzbuchhaltung und Personalsystem verwendet werden. Es sollte also eine Komponente *Organisation* geben, entweder im CaMS oder eine unabhängige, die eng mit dem Personalsystem und der Kostenstellenstruktur gekoppelt ist. Es ist ei-

gentlich selbstverständlich, dass *alle* Organisationsstrukturen in genau *einem* System abgebildet sind und nicht Verwaltung und Fachbereiche auseinander fallen.

Ein anderes Beispiel ist das Gebäudemanagement (*Facility Management*). Wenn ein FM-System betrieben wird, muss dieses bei den Räumen für Lehrveranstaltungen die entsprechenden Attribute pflegen (Medienausstattung, Plätze, Wartungsarbeiten etc.) und sie an das CaMS exportieren. Existiert es nicht, kann es notwendig sein, im CaMS eine Komponente *Raumverwaltung* zu haben, die aber keinesfalls „evolutionär" zu einem Gebäudemanagementsystem ausgebaut werden darf. Im Gegenteil, bei der Beschaffung eines solchen Systems muss diese Komponente des CaMS „rückgebaut" werden.

Durch die Beispiele soll folgender Grundsatz für ein CaMS vermittelt werden:

> Campus-Management Systeme müssen so schlank und modular wie möglich definiert werden, weil der transaktionale Kern allein des Teilsystems *Lehre* schwergewichtig und ein Teilsystem *Forschung* ebenfalls notwendig ist.

2.7 Basisfunktionen und verteilte Grunddaten

In Tabelle 1 auf Seite 6 sind nur anwendungsbezogene Funktionen genannt. Neben allgemein verfügbaren, technischen Basisfunktionen wie Dateitransfer (z. B. herunter laden von Lehrmaterial, herauf laden von Aufgabenlösungen) gibt es vor allem *eine* Basisfunktion, die in Administrativen Softwaresystemen von Wirtschaft und Verwaltung keine auch nur annähernd so wichtige Rolle spielt wie in einem CaMS: Die **Kommunikation** zwischen Lehrenden und Lernenden.

Die Kommunikationsfunktion muss einfach und sicher funktionieren und möglichst unabhängig von der Geschäftspolitik einzelner Hersteller sein, da sie insbesondere im Teilsystem Lehre essenziell ist. Jeder Lehrende muss alle Lernenden oder definierte Untergruppen jeder seiner Veranstaltungen informieren und mit ihnen kommunizieren können. Die Partner dieser Kommunikation wechseln zumindest mit jedem Semester, in vielen Fällen auch laufend (z. B. angemeldete Teilnehmer eines Prüfungstermins). Das CaMS als zentrales System ist dabei der Dreh- und Angelpunkt, da nur hier alle Akteure Flächen deckend erreichbar sind. Insbesondere die Studierenden sollten über das CaMS selbst steuern können, über welches Endgerät diese Kommunikation sie erreichen soll.

Eng verbunden mit der im vorigen Abschnitt besprochenen Organisations-Komponente ist eine **Rechteverwaltung**, die für alle in eine Organisation eingebetteten Anwendungssysteme selbstverständlich ist. Ein guter Teil der Berechtigungen kann dabei aus der Einbettung des Personals in die Organisationsstruktur abgeleitet werden und verringert damit den administrativen Aufwand erheblich.

Eine weitere Basisfunktion ist z. T. technisch begründet. Sie bezieht die originären Daten aus der bereits in Abschnitt 2.6 genannten Personalfunktion. Für fast alle Funktionen in Forschung und Lehre (F&L) wird ein Datentyp ***Person-F&L*** benötigt, dessen zentrale Grunddaten am besten repliziert werden. In den Teilsystemen Lehre, Forschung und Bibliothek wird die *Person-F&L* benötigt, die lehrt, prüft, forscht, publiziert etc. Diese Komponente darf natürlich nur diejenigen Personendaten aus dem zentralen System zeigen, die nicht dem Datenschutz unterliegen und nur jene replizieren, die für die Verarbeitung benötigt werden. Für den Datentyp *Person-F&L* sind dezentral schreibende Zugriffsrechte notwendig, und zwar disjunkt zu den zentralen. Z. B. wird so vermieden, dass eine Person nach der Habilitation im zentralen Personalsystem immer noch als Tutor geführt wird, die Rolle, die vor langer Zeit zur Anlage der Person im zentralen Personalsystem geführt hatte. Hierdurch wird sicher gestellt, dass pro Attribut der *Person-F&L* nur *ein* System führend ist. Zusätzlich kann – wiederum unter dem Aspekt der Kommunikation – die Komponente *Person-F&L* dezentral gepflegte Kontaktinformationen wie Sprechstunden und -orte, sowie bevorzugte Kontaktadressen den Studierenden zur Verfügung stellen.

3 Die Teilsysteme Forschung und "Wissens-Dienste"

Das Teilsystem *Forschung* eines CaMS und der Dienst *Bibliothek* hängen *inhaltlich* stark zusammen, technisch weniger. Dies lässt sich organisatorisch und historisch begründen und hat Auswirkungen darauf, warum *Forschung* zu einem CaMS gehört und *Bibliothek* nicht. Ebenso wird *Controlling* nicht selten als "Kontrolle" wahrgenommen, obwohl es ein Wissens-Dienst ist, der Führungsinformationen aufbereitet. Hierauf geht Abschnitt 3.3 ein.

3.1 Forschung

Für Universitäten werden zwei Hauptaufgaben der in der Institution beschäftigten Personen genannt, Forschung und Lehre. Das ist nur scheinbar trivial. Natürlich sind auch andere Personen beschäftigt, aber sie nehmen Hilfsfunktionen wahr. Die mit den Primäraufgaben betrauten Personen sind in den „leistenden“ Dezentralen (Fakultäten, Fachbereiche etc.) tätig und sollten auch bei ihrer zweiten Hauptaufgabe unterstützt werden, der Forschung. Ein solches Teilsystem gehört allein durch seine Einbindung in die Organisation zu einem CaMS. Dieselben Akteure nehmen beide Hauptaufgaben wahr.

Während jedoch das wichtigste „Produkt“ des Forschungsprozesses traditionell schon lange beachtet wird, die Publikation, weitet sich heute das Interesse an Forschungsinformationen auf Forschungsevaluation und damit auf Metadaten aus ([28, S. 187]), z. B. *Anzahl Publikationen auf A-Level*. Damit werden Vorgangsdaten aus Forschungsprojekten öffentlich relevant, die bisher „privates“ Wissen von Forschern und Instituten waren.

Für Publikationen gibt es bereits ein europäisches Standard-Datenmodell namens *EuroCris* [16]. 'CRIS' steht für *Current Research Information Systems* [1]. Die Vereinigung einiger europäischer Forschungsorganisationen (DFG /DE, JISC /UK, DEFF /DK, SURF /NL) namens *Knowledge Exchange* (KE) gibt relevante Datenbestände für Forschungsinformationssysteme an [40]. Unstrittig ist, dass ein CRIS-System erforderlich ist, welches

- Daten zu Projekten
- Forschungsergebnisse oberhalb der Studiengänge wie Dissertationen und Habilitationen ausweist
- abgeleitete Daten für Evaluationen bereit stellt, etwa die Anzahl der Vorgänge.

Aus den originären, gerade auch beschreibenden Daten müssen sich Berichte generieren lassen.

Wenn Universitäten zunehmend wettbewerbliche Anreize erhalten, müssen sie in der Lage sein, ihre Forschungsleistungen im Verhältnis zu den insgesamt verfügbaren Forschungsressourcen einheitlich und effizient darzustellen. So hat die Universität Münster als fünftgrößte deutsche Hochschule zwar die Komponente Lehre eines CaMS noch zurück gestellt, aber ein Forschungsinformationssystem entwickelt, das schnell einführbar war und hohe Nutzen-Potentiale hat [25]. Auch die FU Berlin berichtet über eine in die dortige SAP-Installation integrierte Forschungs-Datenbank, die aus einer *Drittmittelprojekt-Datenbank* und einer *Profil-Datenbank* für die nicht fremdfinanzierte Forschung besteht. Beide Komponenten haben Schnittstellen zur Publikations-Datenbank der zentralen Bibliothek [35]. Nach Horstmann & Jahn scheint es allerdings in Deutschland nur wenige funktionsfähige Forschungs-Datenbanken zu geben (s. [28, S. 188]).

3.2 Bibliothek und Publikationen

Bibliothekssoftware hat im Gegensatz zu Campus-Management Systemen eine lange Tradition mit speziellen Buchungs-, Archivierungs- und Abfragefunktionen. Es scheinen überwiegend Eigenentwicklungen im Einsatz zu sein, da Standardsysteme offenbar nur für kleine Bibliotheken verfügbar sind ([33; 52]). Ursprünglich wurde nur der Bestand an gedruckten oder in lokalen Datenbanken gehaltenen Inhalten verwaltetet. Für das Erscheinungsbild und die Ressourceneffizienz einer gut profilierten Hochschule ist jedoch heute eine leistungsfähige *Publikations-Datenbank* erforderlich, die

die Zitationskulturen der unterschiedlichen Disziplinen unterstützt. Ob eine Komponente *Publikationen* dem Teilsystem Forschung oder einem System *Bibliothek* zugeordnet wird, ist irrelevant. Wesentlich sind einfache und offene Schnittstellen – insbesondere eine performante und flexible Suche – und Schreibrechte der Erzeuger von Daten aus den Dezentralen. Diese Rechte müssen exklusiv sein, da es sich um Personen bezogene Daten handelt. Es sollte auch keine Publikationspflicht für Forschungsergebnisse geben, wenn die Publikations-Datenbank nicht rechtlich legitimiert und technisch gut abgesichert ist ([25, S. 51]).

Der Service einer Bibliothek für die Lehre, besonders aber für die Forschung, besteht darin, die Mitglieder der Universität schnell (!) mit den Daten zu versorgen, die im Kontext eines Lern- oder Forschungsgegenstands verfügbar sind oder explizit gesucht werden. Zum Dienst gehört auch, die Integrität der Referenzen und Inhalte zu gewährleisten. Die Benutzer möchten weit über die Grenzen der eigenen Institution hinaus Referenzen erhalten, möglichst auch Inhalte, am besten weltweit. Diese Vision scheitert heute noch an gewachsenen Identifikationsstrukturen, sog. Katalognummern (s. [37, S. 364]).

Software und Datenstrukturen aus dem Bibliotheksbereich verlangen spezielle Expertise und sind selbst bei großzügiger Auslegung nicht mehr als Administrative Systeme anzusehen. Nach Neubauer [37] geht die Entwicklung von den Institutionen spezifischen Systemen weg zu ganz wenigen oder nur genau einem Cloud-Katalog. der aber eine weltweit eindeutige Identifikationsnummer für jedes bibliographische Objekt benötigt. Hier konkurrieren Global Players, z. B. die Organisation OCLC (*Online Computer Library Center*) und die Firma Ex Libris.

Für unser Thema dürfte deutlich geworden sein, warum Bibliotheks-Software zwar einfach handhabbare Exportschnittstellen benötigt, aber nicht Teil eines CaMS sein kann. Die Publikations-Datenbank benötigt nur dann Importschnittstellen des Bibliothekssystems, wenn sie in der Forschungskomponente des CaMS gehalten wird.

3.3 Controlling

Die Controlling-Bedürfnisse beim Campus-Management sind vielfältig, nicht erst seit der Einführung von BA/MA und der Budget-Autonomie von Universitäten. Sowohl die Leitungen von Fakultäten als auch von Hochschulen als Ganzem benötigen eine Vielzahl aggregierter Daten oder Kennzahlen (s. hierzu [46] zu den Informationen, die allein ein Dekan benötigt). Zentrales Controlling für eine Hochschule ist mit vertretbarem Aufwand nur möglich, wenn die Fachbereiche semantisch und syntaktisch homogene originäre Daten erzeugen [11]. Dies wiederum ist nur möglich mit einem CaMS, nicht mit einem Flickenteppich von Einzelanwendungen.

Einheitliche Benutzeroberflächen sind zwar hilfreich, aber keine hinreichende Bedingung. So lief eine über Jahre geführte Diskussion der Hochschul-Rechenzentren über die Bedeutung von „Portalen“ (s. z. B. [7]) am Kern des Problems vorbei, der Verdichtungsfähigkeit von Daten.

Nach der vollständigen Einführung des operativen CaMS BIS (Bielefelder Informationssystem) hatte die Universität zwar eine verlässliche Datenbasis fast aller originären Daten, war aber schlecht auskunftfähig [9]. Ein Data Warehouse System, wie es z. B. in der Universität Osnabrück als "Glocke" über HIS-Module gesetzt wurde [41], verbot sich aus verschiedenen Gründen. Mit drei Diplomarbeiten wurde die Vermutung empirisch unterlegt, dass Bearbeiter mit einfachen Programmierkenntnissen mittels eines Frameworks Controlling-Informationen erzeugen können. Dies war für das zentrale Controlling der Universität [11] und die Fakultäten für Soziologie [29] und Wirtschaftswissenschaften [24] in allen drei Fällen erfolgreich.

4 Fazit: Administrative Systeme

Wir haben in den voran gehenden Kapiteln Aussagen gemacht, die als Hypothesen zu grundlegenden Eigenschaften Administrativer Softwaresysteme aufgefasst werden können.

H1. Ein CaMS ist für eine „Big-Bang“ Einführung erheblich zu groß und die Organisation einer Hochschule dafür viel zu komplex.

H2. Ohne Kenntnis der Objekttypen der Grunddaten kann keine nachhaltige Architektur entstehen, die sich iterativ einführen ließe.

H3. Die Generierung abgeleiteter Daten kann durch späteres Einfügen von Methoden der Objektttypen ohne Gefährdung der Datenintegrität geleistet werden.

H4. Gespeicherte abgeleitete Daten sind redundant und ohne Zeitbezug unbrauchbar.

H5. Für wesentliche Teile eines CaMS gelten die Grundsätze ordnungsgemäßer Buchführung (GoB), insbesondere für die Prüfungsverwaltung.

5 Datenmodell eines CaMS

Das Datenmodell eines Administrativen Softwaresystems ist besonders wichtig für das Fachkonzept (s. z. B. [17, S. 189]: „konzeptionelles Objektschema“). Will man ein über die Datenbasis integriertes System evolutionär einführen, also sequentiell, so ist es unerlässlich, wenigstens ein grobes Datenmodell zu kennen. Dadurch lässt sich die Zerlegungsfähigkeit des Systems abschätzen, um funktional benutzbare Komponenten mit sparsamen Schnittstellen zu erhalten. Da fast jede Komponente Vorgangsdaten erzeugt, genügt es, dies für die erforderlichen Grunddaten zu prüfen. Später können Komponenten oder Methoden hinzu kommen, die keine Vorgänge erzeugen, sondern nur abgeleitete Daten, z. B. für das Controlling.

Der Kern eines CaMS-Datenmodells ist die Studienmodellierung, die um nur wenige weitere Grunddaten ergänzt werden muss, um das Datenmodell insgesamt zu erhalten. Wir zeigen ein Grobdatenmodell, eine erste Verfeinerung des zentralen und sehr komplexen Grunddatentyps *Studium*.

5.1 Grobdatenmodell eines CaMS

Das sehr grobe Datenmodell in Abbildung 2 zeigt die (unabhängigen) Grunddaten, die in *Rollen* und *Beziehungstypen* verfeinert werden müssen. Der Typ *Person* ist aus Gründen der Verständlichkeit direkt in die Rollen *Dozent* und *Student* verfeinert.

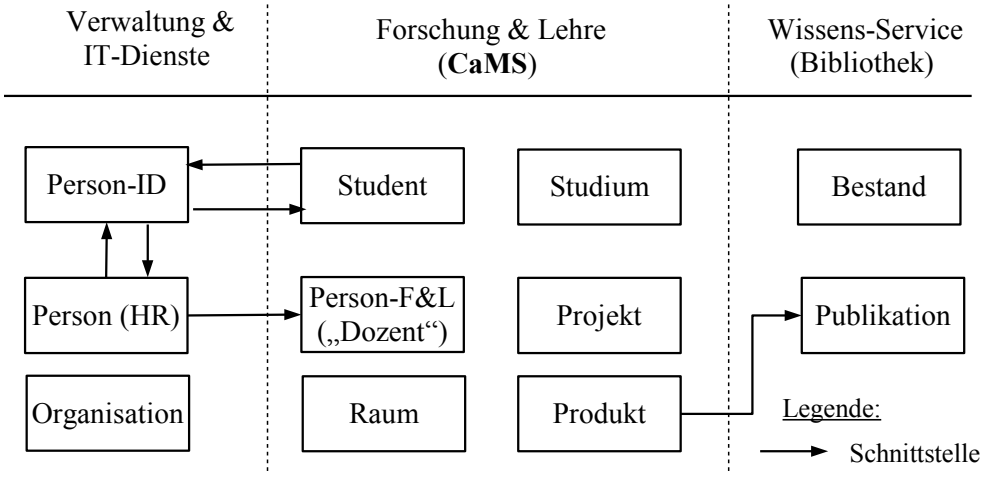

Abb. 2: Grob-Datenmodell der Grunddatentypen mit wichtigen Schnittstellen

Der Datentyp **Person** (*Human Resources*) wird von der Organisationseinheit Personal verantwortet, generiert aber eine Nachricht an das Identity-Management und das CaMS, in welchem die von den Dezentralen gepflegten F&L-Daten des Typs Person ergänzt werden. Der Datentyp **Student** wird ausschließlich vom CaMS verwaltet, muss aber Zugangsdaten über *Person-ID* (ID-Management) erhalten.

Aus Gründen der besseren Übersicht geht Abbildung 2 davon aus, dass kein Gebäudemanagement-System betrieben wird, so dass das CaMS die für die Lehre relevanten Attribute des Objekttyps **Raum** verwaltet.

Der Datentyp **Studium** ist besonders komplex und wird im nächsten Abschnitt verfeinert.

Das Teilsystem Forschung benötigt die Objekttypen **Projekt** und **Produkt**. Die denkbaren Produkte können eng ausgelegt werden; dann zählen nur Publikationen hierzu. Das schließt Abschlussarbeiten aus der Lehre mit ein. Auch Auszeichnungen und Patente werden als Produkte gesehen (Datenmodell in [25]).

Wir nehmen in Abbildung 2 an, dass die **Publikationsdaten** von der Bibliothek verwaltet werden, aber Daten auch aus dem Forschungs-Teilsystem beziehen. Der Kontext des Grunddatentyps **Bestand** wurde in Abschnitt 3.2 besprochen. Dort wurde klargestellt, dass nicht nur der lokale Literaturbestand gemeint ist, sondern der Welt-Wissensbestand zu einem von Forschern bearbeiteten Thema.

Abbildung 3 zeigt nur beispielhaft einige wichtige Vorgangsdaten aus dem Teilsystem *Lehre*, um den Zusammenhang mit den Grunddaten aus Abbildung 2 zu verdeutlichen. Assoziationstypen sind nur angedeutet. Die originären Daten des Prozesses Lehre werden schematisch gezeigt. Die Handhabung von Abschlüssen könnte über eine rudimentäre und übergangsweise manuelle Prüfungsverwaltung funktionieren, wenn die erforderlichen originären Daten existieren.

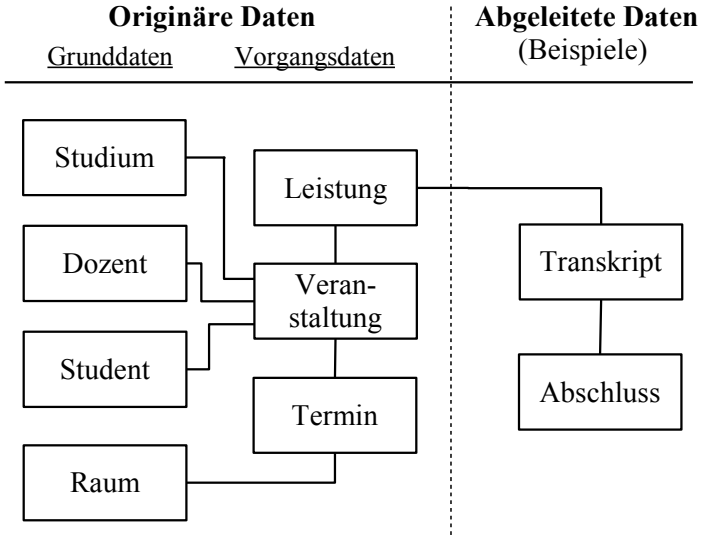

Abb. 3: Ergänzung der Grunddaten durch Vorgangsdaten des Prozesses Lehre

5.2 Der Grunddatentyp Studium

Bisher gibt es keinen publizierten Vorschlag, wie man einen Grunddatentyp *Studium* in der Vielfalt deutschsprachiger Hochschulen allgemein gültig modellieren könnte. [10] ist eine erste Arbeitsversion, die inzwischen weiter entwickelt aber noch nicht veröffentlicht ist. Neubauer spricht schon bei den Bibliothekssystemen von „der deutschen 'Kleinstaaterei'“ ([37, S. 364]). Bei einem CaMS /Lehre stellt sie das Kernproblem dar, das eine Standardisierung der Software verhindert.

Abbildung 4 zeigt eine vereinfachte Übersicht der 11 Objekttypen des Meta-Typs *Gattung (kind)*, die auskonstruiert in weitere 30 *Rollen* und 19 *Beziehungstypen* zerfallen. Das Modell wurde mit der Ontologie UFO (*Unified Foundational Ontology*) [20] unter Verwendung der Methodik von Ortner [38] und der Transformation OntoUML [21] erstellt und mittels einer Stichprobe von 10 der 60 großen deutschsprachigen Hochschulen an zufällig gezogenen 30 Studiengängen validiert. Alle 30 untersuchten Fälle ließen sich mit der in Abbildung 4 gezeigten Referenzstruktur abbilden. Es handelt sich um sieben deutsche, eine österreichische und eine schweizerische Hochschule, sowie eine große deutsche Fachhochschule. Für jede Hochschule wurde eine Stichprobe von 3 Studiengängen gezogen.

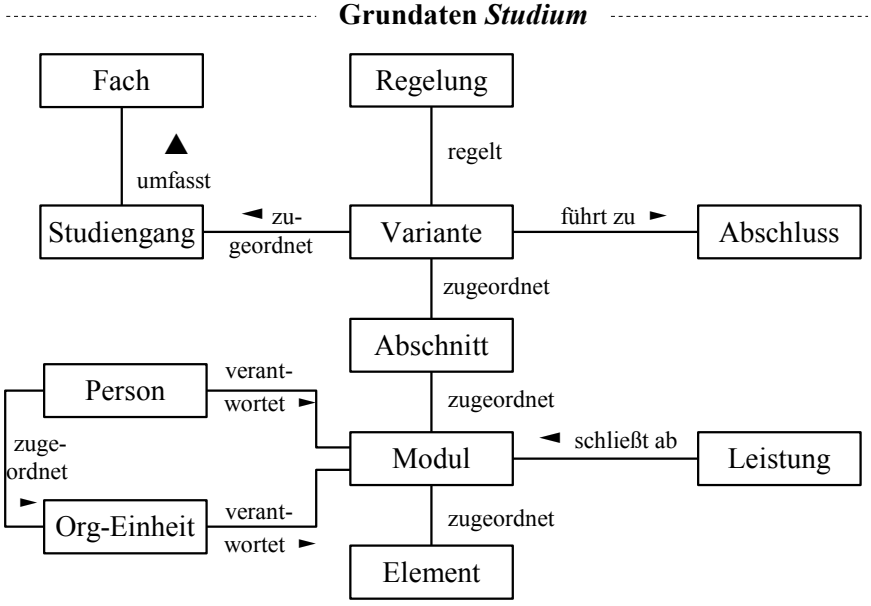

Abb. 4: Grunddatentypen eines Referenzmodells *Studium*

Der Leser beachte, dass in der Modellierung der Abbildung 4 nur die Grunddatentypen des komplexen Datentyps *Studium* enthalten sind. *Person* spricht also zunächst nur die Rolle *Dozent* an. Die Rolle *Student* tritt erst auf, wenn Vorgänge stattfinden. Diese wurden in Abbildung 3 gezeigt.

Die Datentypen in Abbildung 4 haben überwiegend eine Semantik, die den Lesern geläufig sein dürfte. Erklärungsbedürftig erscheinen nur Begriffe der mittleren Spalte. Kennzeichen von BA-/MA-Abschlüssen ist die Variantenvielfalt, die zu Schwerpunkten der Abschlüsse führen kann. Sie tritt beim BA-Studium üblicher Weise nach dem Abschnitt *Kernfach* (meist 4 Semester) in der sog. *Profilphase* auf, in der Profile gewählt werden müssen. In diesem Fall hätte ein BA-Abschluss zwei *Abschnitte*. Der Begriff *Modul* ist gesetzlich vorgegeben, der Begriff *Element* aber eher unüblich. Mit der Aufzählung von nur drei möglichen Ausprägungen von „Element“ dürfte verständlich werden, warum „Veranstaltung“ eine ungeschickte Abstraktion wäre: Element = {Vorlesung, Seminar, …, Praktikum}.

Es scheint gemäß unseren zunächst nur in [10] bekannt gemachten Arbeiten möglich zu sein, ein Referenzmodell zu entwickeln, das BA-/MA-Studiengänge im deutschen Sprachraum abbilden kann. Dies ist für eine zukünftige Standardsoftware sehr wertvoll.

6 Zusammenfassung und Ausblick

Wir haben ausführlich – unter Bezug auf elementare Grundlagen der Datenbanktechnik – die Charakteristika Administrativer Systeme dargelegt. Mit deren Einführung und Betrieb haben Hochschulen, die noch traditionell in "IT = Rechenzentrum" denken, wenig Erfahrung.

Campus-Management Systeme sind wegen der hohen Anzahl zu buchender Vorgänge und der Spezifität der Grunddaten zur Lehre komplexe und damit teure Infrastrukturen. Da Standardsysteme sich erst entwickeln müssen, sind Einführung und Betrieb auch finanziell hoch riskant. Hinzu kommen Anforderungen an eine rechtssichere Archivierung von Daten, die höher sind, als es das HGB für Unternehmen vorschreibt.

Wegen der Integration der Grunddaten besteht ein Campus-Management System aus den Teilsystemen *Lehre* und *Forschung*, sollte aber nicht mit den Diensten einer Bibliothek belastet werden. *E-Learning* hat seinen Schwerpunkt eher im Lernen, weniger in den standardisierten Teilen des Prozesses Lehre und sollte daher ebenfalls, wie *Bibliothek*, von getrennten Systemen unterstützt werden.

Trotz des innovativen Charakters dieser Systeme lässt sich die Datenbasis mit hoher Wahrscheinlichkeit standardisieren. Dies weist den Weg in die Richtung von Standardsoftware. Unseren Erfahrungen nach könnten sich etwa drei Systeme als Standardsoftware etablieren, denn der Markt ist groß und die Nachfrage erheblich. Die heutigen Pilotanwendungen müssen allerdings eine Betriebsphase von mindestens 6 bis 8 Jahren durchlaufen, in der sie zeigen, dass releasefähige Systeme entstanden sind, bei denen sich Kosten und Nutzen mindestens die Waage halten. Dann schreiben wir das Jahr 2020.

Literatur

1 Asserson, A; Keith, J.: Current Research Information Systems (CRIS): Past, Present, Future. Wissenschaftsmanagement 15 (2009) 1, 41-44.

2 Alt, R.; Auth, G.: Campus-Management-System. Wirtschaftsinformatik 52(2010) 3, 185-188.

3 Bick, M.; Grechenig, T.; Spitta, T.: Campus-Management-Systeme. In: *Pietsch, W.; Krams, B.* (Hrsg.): Vom Projekt zum Produkt. Fachtagung Aachen, Dez. 2010, LNI 178, Koellen, Bonn, 61-78.

4 Bittner, S.; Hornbostel, S.; Scholze, F. (Hrsg.): Forschungsinformation in Deutschland: Anforderungen, Stand und Nutzen existierender Forschungsinformationssysteme. Workshop iFQ Berlin, Mai 2012.

5 Bode, A.; Borgeest, R. (Hrsg.): Informationsmanagement in Hochschulen, Springer, Berlin et al. 2010.

6 Borgeest, R.; Pongratz, H.: Austausch universitärer Kernsysteme. In: [Bode et al., 2010], 13-26.

7 Böhm, B.; Held, W.; Tröger, B.: Integriertes Informationsmanagement einer großen Universität. In: Degkwitz, A.; Schirmbacher, P. (Hrsg.): Informationsstrukturen im Wandel. Informationsmanagement an Deutschen Universitäten. Bock & Herrchen, Bad Honnef 2007.

8 Breitner, M.H.; Klages, M.; Sprenger, J.: Wirtschaftlichkeit ausgewählter Campus-Management-Systeme – Auftrag der TU9, Institut für Wirtschaftsinformatik, TU Hannover. http://archiv.iwi.uni-hannover.de/cms/images/stories/upload/lv/sosem10/Systementwicklung/wacamas_finale_v-1_1kurz.pdf (2.5.2013)

9 Brune, H.; Jablonski, M.; Möhle, V.; Spitta, T.; Teßmer, M: Ein Campus-Management-System als evolutionäre Entwicklung. In: [Hansen et al., 2009], 483-492.

10 Brune, H.; Carolla, M.; Spitta, T.: Studiengangsmodellierung – Ein implementierter Diskussionsansatz. In: GI – Rundbrief WI-MAW 35(2013) 1, 14-24. http://pub.uni-bielefeld.de/publication/2670199 (29.04.2014).

11 Carolla, M.: Controlling mit einem Campus Management System – Konzept und Implementierung zur Beurteilung eines Frameworks. Diplomarbeit Fakultät für Wirtschaftswissenschaften, Universität Bielefeld, Nov. 2009.

12 Charette, R.N., Why Software Fails, IEEE Spectrum, 42 (2005) 9, 42-49.

13 Denert, E.: Software-Engineering. Springer, Berlin et al. 1991.

14 Elmasri, R.; Navathe, S.B.: Grundlagen von Datenbanksystemen. 3.Aufl. Pearson, München – Boston et al. 2009.

15 Ernst & Young: Campus-Management zwischen Hochschulautonomie und Bologna-Reform. Studie 2011. http://www.ey.com/Publication/vwLUAssets (24.5.2013).

16 EuroCris: www.eurocris.org (17.10.2013)

17 Ferstl, O.K.; Sinz, E.J.: Grundlagen der Wirtschaftsinformatik. 5. überarb. u. erw. Aufl., Oldenbourg, München – Wien 2009.

18 Fischer, H.; Hartau, C.: STiNE an der Universität Hamburg – Zur Einführung eines integrierten Campus Management Systems. In: [Hansen et al., 2009], 533-542.

19 Grechenig, T.; Bernhart, M.; Breiteneder, R.; Kappel, K.: Softwaretechnik – Mit Fallbeispielen aus realen Entwicklungsprojekten. Pearson Studium, München 2010.

20 Guizzardi, G.; de Almeida Falbo, R.; Guizzardi, R. S. S.: Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The Case of the ODE Software Process Ontology. In: Lencastre, M.; Falcão e Cunha, J.; Valecillo, A. (ed.): 11th Iberoamerican Conference on Software Engineering, Recife, Brasil, 2008, 127-140.
21 Guizzardi, G.; Pinheiro das Gracas, A.; Guizzardi, R. S. S.: Design Patterns and Inductive Modeling Rules to Support the Construction of Ontologically Well-Founded Conceptual Models in OntoUML. In: Salinesi, C.; Pastor, O. (ed.): Advanced Information Systems Engineering Workshops, LNBIP vol. 83, Springer, Berlin et al., 2011, 402-423.
22 Hansen, H.R.; Karagiannis, D.; Fill, H-G. (Hrsg.): Business Services – Konzepte, Technologien, Anwendungen (Bd 2). 9. Int. Tagung Wirtschaftsinformatik, Febr. 2009 Wien.
23 Helmut-Schmidt-Universität Hamburg: Empfehlung für die Auswahl eines Softwaresystems Prüfungsverwaltung / Campusmanagement. Sommer 2006. www.hsu-hh.de/campus-portal/index_V6lDrbV51OIcKbVm.html.
24 Heidemann, S.: Informationen zur Führung einer Fakultät – Erprobung eines Frameworks. Diplomarbeit Fakultät für Wirtschaftswissenschaften, Universität Bielefeld, Jan. 2010.
25 Herwig, S.; Becker, J.: Einführung eines Forschungsinformationssystems an der Westfälischen Wilhelms-Universität Münster – Von der Konzeption bis zur Implementierung. In: [Bittner et al., 2012], 41-53.
26 Herwig, S.; Höllrigl, T.: "All roads lead to Rome": Establishing Best Practices for the Implementation and Introduction of a CRIS: Insights and Experience from a CRIS Project at the University of Münster. In: Jeffery K.; Dvorák, J: (eds.): E-Infrastructures for Research and Innovation: Proc. 11th Int. Conf. on Current Research Information Systems, June 2012, Prague, 93-102.
27 Hochschulrektorenkonferenz (HRK): Für eine Reform der Lehre in den Hochschulen. Beschluss v. 22.4.2008. http://www.hrk-bologna.de/bologna/de/download/dateien/rlehrebeschluss2008.pdf. (3.5.2013)
28 Horstmann, W.; Jahn, N.: Persönliche Publikationslisten als hochschulweiter Dienst – Eine Bestandsaufnahme. BFP 34 (2010), 185-193.
29 Hucke, S.: Unterstützung des Lehrprozesses einer Fakultät durch Führungsinformationen – Informationskonzept und teilweise Implementierung eines Frameworks. Diplomarbeit Fakultät für Wirtschaftswissenschaften, Universität Bielefeld, Mrz. 2010.
30 Janneck, M. et.al.: Von Eisbergen und Supertankern: Topologie eines Campus-Management-Einführungsprozesses. In: [Hansen et al., 2009], 453-462.
31 Klug, H.: Erfolgsfaktoren bei der Umstellung von Informationssystemen an Hochschulen. In: [Hansen et al., 2009], 473-482.
32 Labitzke, S.; Nussbaumer, M.; Hartenstein, H.; Juling, W.: Integriertes Informationsmanagement am KIT – Was bleibt? Was kommt? In: [Bode et al., 2010], 35-46.
33 Lackhoff, M.: Bibliothekssoftware. www.lackhoff.de/biblsoft.html (21.03.2014)
34 Lehman, M.M.: Programs, Life Cycles, and Laws of Software Evolution. Proceedings of the IEEE 68(1980) 9, 1060-1076.
35 Lewerenz, A.: Forschungsdatenbank der Freien Universität Berlin. In: In: [Bittner et al., 2012], 79-89.
36 Mertens, P.: Integrierte Informationsverarbeitung 1 – Operative Systeme in der Industrie. 18.Aufl., Springer/Gabler, Wiesbaden 2013.
37 Neubauer, K.W.: Informationsinfrastruktur für Bibliotheken in Deutschland. b.i.t.online 16 (2013) 5, 363-376.
38 Ortner, E.; Söllner, B.: Semantische Datenmodellierung nach der Objekttypenmethode. Informatik Spektrum 12(1989) 1, 82-92.
39 Radenbach, W.: Integriertes Campus Management durch Verknüpfung spezialisierter Standardsoftware. In: [Hansen et al., 2009], 503-512.
40 Razum, M.; Simons, E.; Horstmann, W.: Exchanging Research Information. http://driver-repository.be/media/docs/KEIRstrandreportExchangingResearchInfoFINALFeb07.pdf (19.10.2013)
41 Rieger, B.; Haarmann, T.; Höckmann, E.; Lüttecke, S.: Data Warehouse basierte Entscheidungsunterstützung für das Campus-Management an Hochschulen. In: [Hansen et al., 2009], 523-532.
42 Schäfer, J. P.; Grauer, M. (Hrsg.): Universitätsverwaltung und Wirtschaftsinformatik. Proceedings, Siegen, Okt. 1995.
43 Sinz, E., Krumbiegel, J.: Gestaltung qualitätsgesicherter Universitätsprozesse am Beispiel des Prozesses ‘Lehre und Studium’. In: [Schäfer & Grauer, 1995], 15-32.
44 Spitta, T.: Software Engineering und Prototyping – Eine Konstruktionslehre für administrative Softwaresysteme. Springer, Berlin et al. 1989.
45 Spitta, T.; Mordau, J.: Entwicklung und Ergebnisse eines allgemeingültigen Fachkonzeptes für die Prüfungsverwaltung an Hochschulen. In: [Schäfer & Grauer, 1995], 128-147.
46 Spitta, T.: Standardsoftware zur Verwaltung und Führung von Fakultäten. Eingeladener Vortrag GI `97, Aachen. In: Univ. Bielefeld, Fakultät für Wirtschaftswissenschaften, Diskussionspapier 354, August 1997. http://pub.uni-bielefeld.de/publication/2669574 (29.04.2014)
47 Spitta T.: Sechs Jahre Anwendungsentwicklung mit Prototyping – Revision von Begriffen und Konzepten. In: Züllighoven H.; Altmann W.; Doberkat E.-E. (Hrsg.): Requirements Engineering ´93, Berichte des German Chapter of the ACM 41, Teubner, Stuttgart 1993, 49–66.
48 Spitta, T.; Bick, M.: Informationswirtschaft. 2. Aufl., Springer, Berlin et al. 2008.
49 Strobl, S.; Bernhart, M.; Grechenig, T.; Kleinert, W.: Digging Deep: Software Reengineering supported by Database Reverse Engineering of a System with 30+ Years of Legacy. IEEE Conf. on Software Maintenance 2009, Edmonton, Canada, 407-410.
50 TU München: Hintergrundinformationen zum Projekt CM@TUM vom 19.02.2008 https://portal.mytum.de/iuk/cm/dokumente/00.allgemein (02.05.2013)
51 TU Wien: Forschungsportal. https://tiss.tuwien.ac.at/research/main.xhtml. (30.09.2013)
52 UB Braunschweig: Bibliotheks-Datenbank. www.allegro-c.de (21.03.2014)
53 Vetter, M.: Aufbau betrieblicher Informationssysteme mittels konzeptioneller Datenmodellierung, 6. Aufl., Teubner, Stuttgart 1990.
54 Vossen, G.: Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme. 5. Aufl. Oldenbourg, München – Wien 2008.
55 Wikipedia: Hochschulinformationssystem. http://de.wikipedia.org/wiki/Hochschulinformationssystem (3.5.2013)
56 Wirtschaftsinformatik-Lexikon: Kurbel / Becker / Gronau / Sinz / Suhl (Hrsg.): Enzyklopadie der Wirtschaftsinformatik. http://www.enzyklopaedie-der-wirtschaftsinformatik.de (26.5.2013)